

Lab 2: .NET 3.5 Graphical Application Client for caBIO caGrid Service

caBIG 2009 Annual Meeting Hack-a-thon

University of Virginia eScience Group

Marty Humphrey, Director

Overview: Create a graphical .NET application as a client for the caBIO caGrid Service. We use the Visual Studio capabilities for designing a GUI –Graphical User Interface – that includes commonly-used controls in Windows applications. A graphical wizard built into Visual Studio is used to build client-side proxy objects for the caBIO caGrid Service. We then add code that will generate the CQL query and display the results in a table.

NOTE: *This lab requires that you performed Lab 1 and have the results of that lab available for use in this lab (specifically “caBioSvc.cs” and “app.config”).*

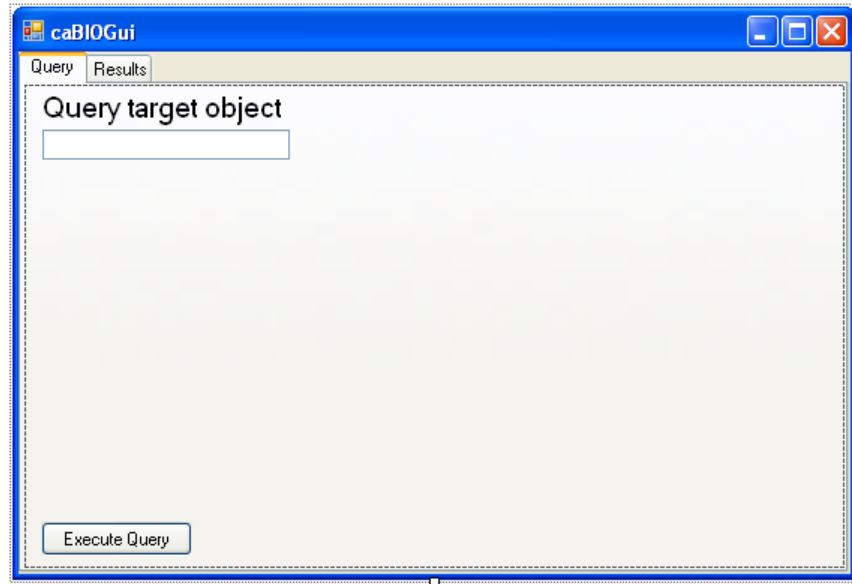
Expected duration: 30 minutes

Part 1: Simple GUI for caGrid Service

1. The first step is to create the Visual Studio project. Run Visual Studio 2008 (Start → All Programs → Microsoft Visual C# 2008 Express Edition) and create a new C# project: Windows Forms Applications app (File → New Project → Windows Form Application). Name it “caBIOGui” and hit “OK” in this dialogue window.
2. Our first task is to add some user controls to the empty window “Form1” and name them. The components can be dragged and dropped from the Toolbox. The Toolbox is located on the left part of the screen (if it is not visible, click on View → ToolBox).
 - a. Unfold the Containers category in the Toolbox and drag and drop a TabControl on the empty form. With the TabControl you just created selected, right click and select Properties. From the list of properties, select “Dock” and change the value to “Fill” –represented by the big square in the middle (clicking on “None” will show you a list of options). This will expand the TabControl to the size of the Form. If you do not see the “Dock” property make sure that the whole TabControl is selected, not just one of the individual tabs.
 - b. We can change some of the displayed names of the controls:
 - i. Select the Form by clicking on the title bar and change the property “Text” to “caBIOGui”.
 - ii. Click on tabPage1, then click on the middle of the tab and change the property “Text” to “Query”.
 - iii. Similarly, change the name of tabPage2 to “Results”.
 - c. Select the first tab (“Query”), (*make sure you have “Query” selected by looking at the “Text” property in the lower-right of Visual Studio, which should say “Query”*), expand “Common Controls” in the Toolbox and drag and drop the following controls to the tab: Button, Label and TextBox (just place them anywhere on the “Query” space in the Windows Form for the moment). Modify the following properties:
 - i. For the Label: “Text” to “Query Target Object”, “Font” to 14 point regular Microsoft Sans Serif.
 - ii. For the TextBox, set “(Name)” to “targetTB”, and “Text” to “gov.nih.nci.cabio.domain.Taxon”. *Note: (“(Name)”) can be found toward the top of*

the list of properties), and "Taxon" will be the default object but can be changed when the user runs the application)

- iii. For the Button, set "Text" to "Execute Query" and resize it to show both words.
- d. All the controls should be resized and moved around to match approximately the following layout (note: your "textbox" should show "gov.nih.nci.cabio.domain.Taxon")



- e. The last step in the GUI design for this simple application is to add a control to display the results of the query. Select the tab "Results" and from the Toolbox drag and drop a DataGridView control (under the "Data" category). Deselect the checkboxes "Enable Adding", "Enable Editing", "Enable Deleting" and click on "Dock in parent container". Change the property "(Name)" to "resultsGrid".

Part 2: Adding the caGrid Service reference

- 3. Our second task is to generate the client-side proxy objects and configure the service endpoint. This is essentially what we did in Lab 1, so we're just going to re-use those results:
 - a. Right-click on caBIOGui project in the Project Explorer -- it's the line immediately below the "Solution 'caBIOGui' (1 solution)", select "Add → Existing Item" and navigate to "caBIOClient" folder from lab1 then select caBioSvc.cs
 - b. Do this "Add → Existing Item" again, change the dialogue's "Objects of type" to be "All Files", and add "app.config"
 - c. We now need to add the appropriate libraries to the solution (we didn't have to do this in Lab 1 because in Lab 1 we did "Add Service Reference", which adds them automatically):
 - i. Right-click on caBIOGui project in the Project Explorer, "Add Reference" and select "System.Runtime.Serialization"
 - ii. Do the same for "System.ServiceModel"
 - d. Just to make sure everything is okay, compile everything by selecting "Build → Rebuild Solution" (don't run the application yet)

Part 3: Adding client code to the GUI

4. Our third task is to add the necessary code to generate the CQL query based on the user input, get the response from the server and display the results on the “DataGridView” control:
 - a. Double click on “Form1.cs” in the “Solution Explorer” or select the tab “Form1.cs [Design]” to display the GUI.
 - b. Double click on the button “Execute Query”. Visual Studio will open the C# code file “Form1.cs” and create a new method “button1_Click”. This method will be executed each time the user clicks on the button. Copy the following code within “button1_Click”:

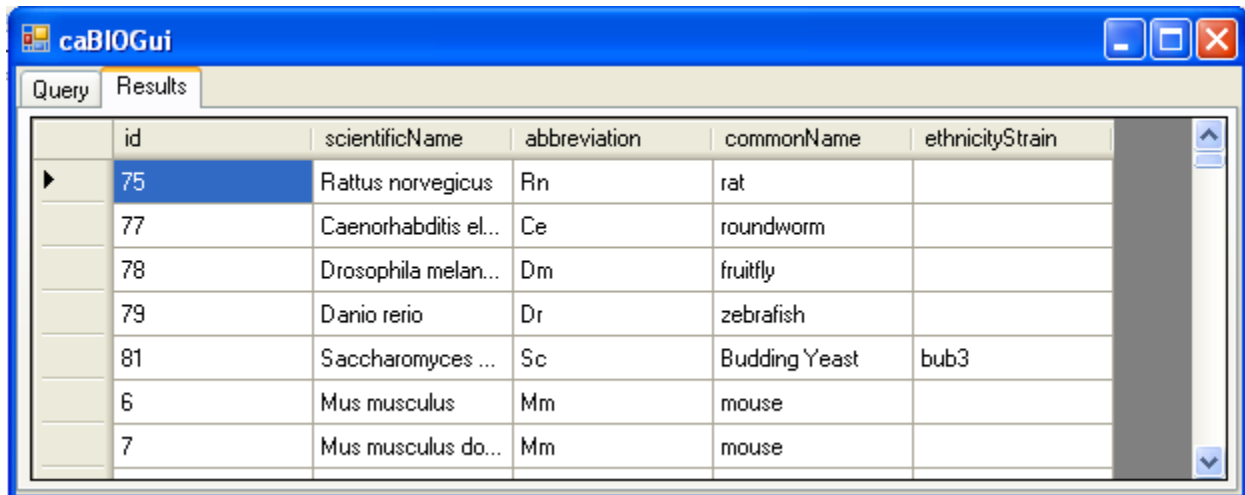
```
CaBIO40GridSvcPortTypeClient proxy = new CaBIO40GridSvcPortTypeClient();
QueryRequestCqlQuery arg = new QueryRequestCqlQuery();
arg.CQLQuery = new CQLQuery();
arg.CQLQuery.Target = new Object();
arg.CQLQuery.Target.name = targetTB.Text;
CQLQueryResults result = proxy.query(arg);

if (result.Items != null)
{
    if (result.Items[0].GetType().FullName.Equals("CQLObjectResult"))
        resultsGrid.DataSource = processCQLObjectResult(result);
}
```

- c. Finally, add the method “processCQLObjectResult” to the class “Form1” by copying the following code after the “button1_Click” method. This is the code that essentially maps the XML returned from the service into the table on the “Results” pane.

```
internal object processCQLObjectResult(CQLQueryResults response)
{
    DataTable table = new DataTable();
    foreach (CQLObjectResult result in response.Items)
    {
        DataRow dr = table.NewRow();
        foreach (System.Xml.XmlAttribute attr in result.Any.Attributes)
        {
            if (!attr.Name.Contains("xmlns"))
            {
                if (!table.Columns.Contains(attr.Name))
                {
                    DataColumn dc = new DataColumn();
                    dc.DataType = System.Type.GetType("System.String");
                    dc.ColumnName = attr.Name;
                    table.Columns.Add(dc);
                }
                dr[attr.Name] = attr.Value;
            }
        }
        table.Rows.Add(dr);
    }
    return table;
}
```

5. Save the project by pressing Ctrl-S and hit F5 to execute. Hit the “Execute Query” button, select the “Results tab (and then wait a few moments while the client retrieves results from the service). If successful, you will see a window similar to the screenshot shown below. Switch back to the “Query” tab, fill the text box with a different value (e.g., “gov.nih.nci.cabio.domain.Chromosome”) and click on the button “Execute Query”. After a couple of seconds, click on the tab “Results” and you will see a table of the results. *Note that everything is re-sizable, and furthermore that the results can be sorted by clicking on the particular column header.*



The screenshot shows the caBIOGui application window. It has a blue title bar with the text "caBIOGui" and standard window control buttons. Below the title bar are two tabs: "Query" and "Results". The "Results" tab is active, displaying a table with the following columns: "id", "scientificName", "abbreviation", "commonName", and "ethnicityStrain". The table contains seven rows of data. The first row is highlighted with a blue background.

	id	scientificName	abbreviation	commonName	ethnicityStrain
▶	75	Rattus norvegicus	Rn	rat	
	77	Caenorhabditis el...	Ce	roundworm	
	78	Drosophila melan...	Dm	fruitfly	
	79	Danio rerio	Dr	zebrafish	
	81	Saccharomyces ...	Sc	Budding Yeast	bub3
	6	Mus musculus	Mm	mouse	
	7	Mus musculus do...	Mm	mouse	